Risk-Aware Model-Based Offline Reinforcement Learning via Bellman Risk Propagation

Zhilin Zhang[1,2],Zihao Li[1,2],Xiaoyu Shi[1,2,*],Yun Lu[1,2],Yanan Bai[3],Zepeng Gong[3]
[1]*Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences*, Chongqing, China
[2]*Chongqing School, University of Chinese Academy of Sciences*, Chongqing, China
[3]*Chongqing University of Technology*, Chongqing, China
[*]Corresponding author:xiaoyushi@cigit.ac.cn

***Abstract*—Offline reinforcement learning (RL) learns policies from fixed datasets without environment interaction, improving safety and sample efficiency. However, distributional shift between behavior and learned policies often causes severe value overestimation, especially for out-of-distribution actions. Existing conservative methods mitigate this by restricting policy updates within dataset support, but their excessive conservatism limits policy improvement. Alternatively, data augmentation and model-based approaches expand the training distribution yet lack principled control over risks from unreliable synthesized samples.In this paper, we propose Model-Based Risk-Aware Policy Optimization (RAPO), a framework that explicitly models long-horizon decision risk to address overestimation in offline RL. RAPO uses a learned dynamics model to generate interactive samples and introduces a dedicated risk value function that propagates extrapolation risk via Bellman backups. This decouples risk estimation from reward learning, enabling transparent and adaptive risk-aware policy optimization. We further design a risk-adaptive weighting mechanism that balances exploration and conservatism based on policy behavior and dataset quality. We theoretically show that RAPO achieves a tighter performance lower bound than existing offline RL methods. Empirical results on diverse D4RL benchmarks demonstrate consistent and significant improvements over state-of-the-art baselines.**

***Index Terms*—Offline Reinforcement Learning, Distribution Shift, Model-Based Policy Optimization, Extrapolation Risk, Risk-Aware Learning**

I. Introduction

Offline reinforcement learning (offline RL) learns policies from fixed datasets without environment interaction [1]. This paradigm is attractive in safety-critical or high-cost domains such as recommender systems [2] and robotic control [3], where online exploration is impractical or unsafe.

Despite its promise, learned policies often overestimate out-of-distribution (OOD) actions due to distributional shift between the behavior and target policies, leading to suboptimal performance. Standard value-based updates rely on Bellman backups with a maximization operator, favoring actions with spuriously high predicted values. Without online interaction, these errors compound over successive updates, causing unstable training and degraded policies [4, 5]. Importantly, this reflects a fundamental problem: *the learned policy lacks reliable signals for how much its decisions should be trusted beyond the data support*.

A dominant line of research addresses overestimation through conservative offline RL methods, which constrain policy updates to stay close to the behavior policy [5, 6, 7]. While these algorithms mitigate overestimation, they restrict policy improvement, especially when the dataset is suboptimal or incomplete [8].

Another direction expands data coverage through augmentation [9, 10]. By synthesizing additional transitions, these approaches broaden the effective support of the offline dataset and enable more extensive policy exploration. However, the reliability of generated samples is difficult to guarantee, and model-induced errors can introduce new bias. Moreover, such methods still rely on conservative strategies to prevent revisiting OOD actions [11].

Model-based offline RL methods offer principled data expansion by learning an explicit dynamics model for virtual interactions [12, 13]. Among them, MOPO [12] penalizes rewards based on model uncertainty to discourage exploitation of unreliable regions. While effective, this design tightly couples uncertainty estimation with reward shaping, making the risk–reward trade-off sensitive to manually tuned penalty coefficients. Furthermore, step-wise uncertainty penalties fail to capture the long-horizon, policy-dependent consequences of risky decisions, limiting their expressiveness in complex offline settings.

In this work, we argue that overestimation in offline RL stems from the lack of **explicit long-horizon decision risk modeling**. To address this, we propose **Model-Based Risk-Aware Policy Optimization (RAPO)**, a framework that decouples risk estimation from reward learning via a dedicated risk value function. Instead of treating uncertainty as an immediate reward penalty, RAPO propagates extrapolation risk through Bellman backups to estimate the cumulative decision risk of a policy. This provides a transparent mechanism for guiding policy optimization away from unreliable regions while preserving improvement capacity.

RAPO leverages a learned dynamics model to generate interactive samples and uses model uncertainty as a proxy for extrapolation risk. The risk signal is learned through a separate risk value function, which estimates long-term risk

This work is supported in part by Key Project of Chongqing Natural Science Foundation(No. CSTB2025NSCQLZX0061), in part by the Science and Technology Innovation Key R&D Program of Chongqing(CSTB2025TIADSTX0023, CSTB2023TIADSTX0031), in part by Chongqing University of Technology Undergraduate Education and Teaching Reform Research Project (General Project)(Project No.: 2024YB76).

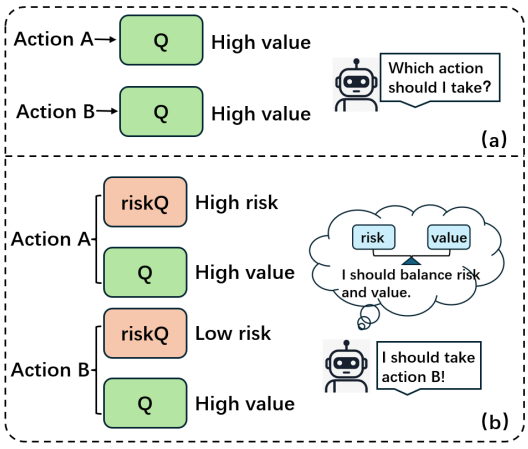

Fig. 1. Traditional offline RL relies solely on Q-values for decision-making, which can lead to suboptimal policies when distribution shift causes overestimation. RAPO introduces riskQ to avoid states with incorrectly estimated values via explicit long-horizon decision risk modeling.

analogously to how standard value functions estimate expected return. To adapt to offline datasets of varying quality, we further introduce a risk-adaptive weighting mechanism that dynamically balances exploration and risk avoidance based on policy behavior and data coverage, eliminating manual tuning.

We theoretically show that RAPO achieves a tighter performance lower bound than existing offline RL methods. Extensive experiments on D4RL benchmarks across multiple domains and dataset qualities demonstrate that RAPO consistently outperforms state-of-the-art baselines with stable learning behavior. In summary, our main contributions are:

- A risk-aware framework that models long-term decision risk by propagating extrapolation uncertainty as a cumulative risk signal via Bellman updates. This reframes value overestimation as a failure to account for risk, moving beyond simple conservatism or reward penalties.
- A risk-adaptive weight that dynamically adjusts the influence of riskQ based on policy exploration level and offline data quality, enabling safe exploration while guarding against erroneous data.
- Extensive experiments on diverse D4RL benchmarks show that RAPO consistently outperforms state-of-the-art offline RL baselines with stable learning, validating explicit risk modeling in offline RL. Code is available at *https://github.com/EmiliatanGG/RAPO*.

II. Preliminaries

Markov Decision Process(MDP).A standard MDP is defined as a tuple:$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \mu_0, \gamma)$,where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(s' \mid s, a)$is the state transition probability distribution, $r(s, a)$ is the reward function, μ_0 is the initial state distribution, and $\gamma \in (0, 1)$ is the discount factor.The policy $\pi(a|s)$ defines the probability distribution of selecting action a given state s.The performance of $\pi(a|s)$ is measured by the expected discounted return $J_{\mathcal{M}}(\pi) = \mathbb{E}_{\pi,P}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$. The optimal policy π^* satisfies $J_{\mathcal{M}}(\pi^*) \geq J_{\mathcal{M}}(\pi)$ for any policy π.The action-value function Q^π is used to measure the expected cumulative reward obtained by taking action a in state s and then following policy π thereafter.It is fitted based on the following Bellman equation:

$$Q^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a), a' \sim \pi(\cdot|s')}[Q^\pi_{tar}(s', a')]. \tag{1}$$

Soft Actor-Critic (SAC). SAC encourages exploration by maximizing the weighted sum of cumulative rewards and policy entropy. The policy optimization objective (Actor Loss) in practice is typically formalized as:

$$\mathcal{L}_{\text{actor}}(\pi) = \mathbb{E}_{s \sim \mathcal{D}}\left[\alpha \log \pi(a|s) - \min(Q_{\theta_1}(s, a), Q_{\theta_2}(s, a))\right], \tag{2}$$

where $a \sim \pi(\cdot|s)$, Q_1 and Q_2 are two action-value functions trained simultaneously to mitigate overestimation of values.

III. RAPO:Risk-Aware Policy Optimization

We train a dynamics model on offline data as a surrogate environment, analogous to MOPO. However, model-based exploitation alone cannot resolve the out-of-distribution challenge. The core issue is balancing exploration and conservatism: we aim to uncover high-performing behaviors via the surrogate model while avoiding overconfident reliance on uncertain predictions. To address this, we quantify the risk of over-exploring high-uncertainty areas, develop a mechanism for adaptively trading off return maximization against estimated risk, and derive a performance lower bound that guarantees worst-case return. The following sections elaborate on each step.

A. Quantifying Exploration Risk

To comprehensively quantify the risk of exploration, we first require a dynamics model capable of both making predictions and quantifying their associated uncertainty. Following MOPO, we construct an ensemble of stochastic neural networks, each parameterizing a Gaussian distribution over next-state–reward pairs given a state–action input:

$$\hat{T}_{\theta,\phi}(s_{t+1}, r|s_t, a_t) = \mathcal{N}(\mu_\theta(s_t, a_t), \Sigma_\phi(s_t, a_t)), \tag{3}$$

Each model is independently trained via maximum likelihood estimation. The uncertainty predicted by the entire ensemble model is characterized by:

$$u(s, a) = \max_{i=1,\ldots,N} \|\Sigma^i_\phi(s, a)\|_F, \tag{4}$$

We use the maximum, rather than the average, to quantify uncertainty. This provides a more conservative and reliable risk estimate for each decision step. Moreover, since the offline dataset consists of ground-truth raw data, the uncertainty for all data points is defined as 0.

However, merely balancing exploration and exploitation based on the risk of a single decision step is insufficient. This is because a decision step with low risk might transition the system to a state where all subsequent decisions are highly risky. Inspired by the idea of approximating the Q-value via the Bellman equation in reinforcement learning to quantify

the long-term reward of an action(Eq.1), we approximate the long-term risk, termed as $riskQ$, using the following equation:

$$riskQ(s,a) \leftarrow u(s,a)+ \gamma \cdot riskQ_{tar}\left(s', \arg\min_{a'} riskQ_{tar}(s',a')\right), \quad (5)$$

Here, the $argmin$ operation encourages the policy to prefer actions that minimize long-term risk. However, in the offline setting, the input (s', a') to $riskQ$ often lies out-of-distribution, and single-network estimation can be affected by extrapolation noise and function approximation errors, leading to systematic underestimation of the risk value (i.e., being overly optimistic about out-of-distribution regions)[14, 15]. To address this, we introduce double $riskQ$-learning: we maintain two independently parameterized risk critic networks $\{riskQ_{\theta_1}, riskQ_{\theta_2}\}$. During the backup process, we first take the minimum $riskQ$ action within each network and then take the maximum estimate between the two networks to achieve a pessimistic risk assessment for out-of-distribution regions:

$$riskQ(s,a) \leftarrow u(s,a)+ \gamma \cdot \max_{i\in\{1,2\}} riskQ^i_{tar}\left(s', \arg\min_{a'} riskQ^i_{tar}(s',a')\right), \quad (6)$$

This design preserves the recursive Bellman structure while suppressing bias in single estimations. Risk critic target networks slowly track main network parameters via delayed soft updates, reducing error accumulation from distribution shift and ensuring stable long-term risk estimation.

B. Risk-adaptive Weight

Using the methods above, we quantify long-term risk at each decision step, reducing the exploration–conservatism trade-off to balancing Q and $riskQ$. Since rewards and risks may differ in magnitude, normalization is required. The first normalization is applied before initial training on model-generated data; subsequent data follows the same parameters, avoiding fitting errors due to inconsistent scaling.

With reward and risk on the same scale, policy exploration is controlled by adjusting the coefficients of Q and $riskQ$ in the actor loss. Data from policy–model interaction before each training round reflects the policy's exploration level. Based on this, we design a risk-based coefficient $\beta(p)$ to automatically balance exploration and exploitation:

$$\beta(p) = \beta_0 + (\beta_1 - \beta_0)\cdot p, \quad (p = \frac{\mu_u}{\mu_u + \sigma_u} \in [0,1]) \quad (7)$$

where μ_u and σ_u denote the mean and variance of the risk from data generated by the interaction between the current policy and the dynamics model, respectively. β_0 and β_1 represent the minimum and maximum weights, respectively.

Intuitively, when risk is high, the policy is over-exploring and should be more conservative, preferring actions with lower $riskQ$. Conversely, when risk is low, we can afford more exploration. Thus, the actor loss is defined as:

$$\text{actor loss} = \alpha \log \pi(a|s) - \left(\min(Q^1, Q^2) + \beta(p)\cdot \max(riskQ^1, riskQ^2)\right). \quad (8)$$

C. Algorithm Performance Analysis

The core of the RAPO algorithm lies in its policy optimization objective:

$$J_{\text{RAPO}}(\pi) = \mathbb{E}_{(s,a)\sim\rho^\pi_{\hat{T}}}[Q(s,a)] - \beta(p)\cdot\mathbb{E}_{(s,a)\sim\rho^\pi_{\hat{T}}}[\text{risk}Q(s,a)], \quad (9)$$

Based on Lemma 4.1 in MOPO[12] and the subsequent derivations (especially Equations (2) and (7)), it can be concluded that for the policy π fitted by MOPO, there exists a lower bound on its true return:

$$\eta_M(\pi) \geq \mathbb{E}_{(s,a)\sim\rho^\pi_{\hat{T}}}[r(s,a)] - \lambda\cdot\epsilon_u(\pi), \quad (10)$$

where $\epsilon_u(\pi) = \bar{\mathbb{E}}_{\rho^\pi_{\hat{T}}}[u(s,a)]$ is the average uncertainty of policy π over the model trajectories, and λ is a constant related to the discount factor γ and the range of the value function.

In our framework, instead of directly using the single-step uncertainty $u(s,a)$ as the penalty term, we introduce the long-term risk estimator $riskQ(s,a)$. This function is defined as the cumulative discounted risk with $u(s,a)$ as the risk:

$$riskQ^\pi(s,a) = \mathbb{E}_{\hat{T},\pi}\left[\sum_{t=0}^{\infty}\gamma^t u(s_t,a_t) \mid s_0 = s, a_0 = a\right], \quad (11)$$

So we can obtain:

$$\mathbb{E}_{(s,a)\sim\rho^\pi_{\hat{P}}}[\text{riskQ}(s,a)] = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{(s_t,a_t)\sim\rho^\pi_{\hat{P},t}}[u(s_t,a_t)], \quad (12)$$

Recognizing that:

$$\epsilon_u(\pi) = \mathbb{E}_{\rho^\pi_{\hat{P}}}[u(s,a)] = \gamma\sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{\rho^\pi_{\hat{P},t}}[u(s_t,a_t)]. \quad (13)$$

Therefore:

$$\mathbb{E}_{\rho^\pi_{\hat{P}}}[\text{riskQ}(s,a)] = \frac{1}{1-\gamma}\epsilon_u(\pi), \quad (14)$$

This indicates that $\mathbb{E}[\text{riskQ}]$ is an effective proxy for the cumulative model error $\epsilon_u(\pi)$.

Substituting the above relationship into the lower bound of MOPO, we obtain:

$$\eta_M(\pi) \gtrapprox \mathbb{E}_{(s,a)\sim\rho^\pi_T}[r(s,a)] - \frac{\beta(p)}{1-\gamma}\epsilon_u(\pi), \quad (15)$$

Note that $\mathbb{E}_{\rho^\pi_T}[r(s,a)]$ and $\mathbb{E}_{\rho^\pi_{\hat{T}}[Q(s,a)]}$ are highly correlated in the sense of expectation. Therefore, the optimization objective $J_{\text{RAPO}}(\pi)$ of RAPO can be regarded as directly maximizing a tight lower bound of the true return $\eta_M(\pi)$. The adaptive weight $\beta(p)$ dynamically adjusts the penalty coefficient in the lower bound in practice, to achieve a balance between exploration and exploitation.When $\beta(p) = \lambda(1-\gamma)$, RAPO achieves a performance lower bound equivalent to that of MOPO. Since $\lambda(1-\gamma)$ is a hyperparameter, while $\beta(p)$ can adaptively adjust according to the environment, RAPO may obtain a tighter lower bound in practice.

A key feature of RAPO is its explicit safety guarantees for the policy under extreme parameter settings.

(Safety Lower Bound) When the adaptive weight $\beta(p)$ is sufficiently large, the policy optimization objective of RAPO degenerates to:

$$\pi \rightarrow \arg\min_{\pi} \mathbb{E}_{(s,a)\sim\rho^{\pi}_{\hat{T}}} \left[\text{riskQ}(s,a)\right]. \tag{16}$$

Given that the risk of data in the offline dataset is 0, it follows that $riskQ \approx 0$. Therefore, the policy that minimizes the long-term risk will converge to a distribution highly similar to the behavioral policy π_B, that is:

$$\pi(a|s) \approx \pi_B(a|s), \tag{17}$$

In this case, the performance lower bound of the learned policy is the performance of behavior cloning:

$$\eta_M(\pi) \geq \eta_M(\pi_B) - \epsilon, \tag{18}$$

where ϵ represents the minor performance gap caused by function approximation and other factors.

Combining Eq.15 and Eq.18, we can provide an overall performance guarantee for RAPO.

(Composite Performance Lower Bound) The final policy π learned by the RAPO algorithm satisfies the following performance lower bound:

$$\eta_{\mathcal{M}}(\pi) \geq \max\Big\{ \sup_{\pi} \big\{\mathbb{E}_{\rho^{\pi}_{\hat{P}}}[r(s,a)] - \frac{\beta(p)}{1-\gamma}\,\epsilon_u(\pi)\big\}, \eta_{\mathcal{M}}(\pi_B)\Big\} - \epsilon \tag{19}$$

This lower bound shows RAPO achieves strong generalization—its adaptive mechanism yields performance comparable to MOPO, while its risk-aware mechanism guarantees performance no worse than behavior cloning. Thus, RAPO adapts to offline datasets of varying quality and risk preferences.

IV. Experiments

In this section, we address the following questions experimentally:

- **RQ.1** How does RAPO perform compared to existing offline reinforcement learning algorithms?
- **RQ.2** Is the performance gain attributable to the risk-aware mechanism and the risk-adaptive weighting scheme?
- **RQ.3** To what extent do the number of policy–model interaction steps and the risk weight influence the algorithm's performance?

We dedicate three subsections to answering these questions in sequence.

A. Evaluation on the D4RL Benchmark(RQ.1)

We evaluate RAPO against several offline RL baselines on D4RL [16], an open-source platform with standardized datasets and evaluation protocols across diverse real-world scenarios. Our experiments cover 12 offline settings spanning three tasks and four dataset types:

- **Random**: 1M samples collected by a random policy.
- **Medium**: 1M samples from a partially trained SAC agent.

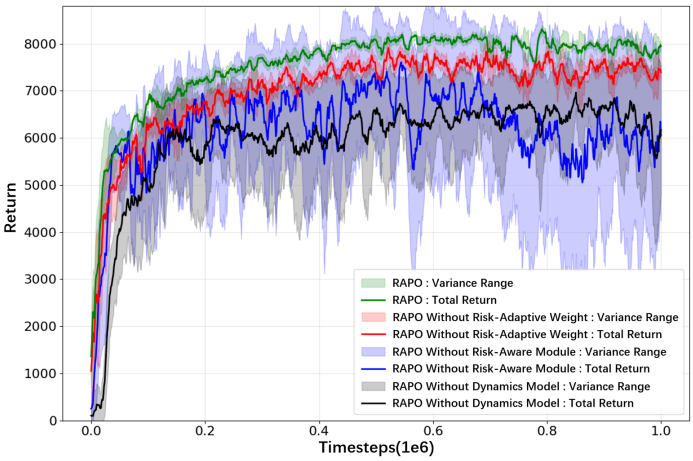

Fig. 2. **Ablation Studies.** Results on the halfcheetah-medium-replay environment are shown above. We sequentially ablate three core components: the risk-adaptive weight, the risk-aware mechanism, and the dynamics model. In the figure, green represents the full RAPO algorithm; red indicates removal of the risk-adaptive weight; blue shows removal of the risk-aware mechanism (including the weight); and black denotes removal of the dynamics model.

- **Medium-Replay**: Replay buffer collected up to the point when the agent reached medium performance.
- **Medium-Expert**: A balanced mix of expert and suboptimal trajectories.

Baseline algorithms include conservative policy learning methods (BC(Behavior Cloning), CQL[5], TD3+BC[17]), data augmentation approaches (SYNTHER[18], PGD[19]), and model-based algorithms (MBPO[20], MOPO[12]).

The experimental results are shown in Table 1.The RAPO method achieves the best performance on all Random and Medium-Replay datasets and also performs best on most Medium datasets. On the Medium-Expert dataset, the RAPO algorithm also achieves competitive performance. In contrast, the MBPO and MOPO algorithms perform poorly, which further supports the conclusions of our performance analysis.

B. Ablation Studies(RQ.2)

To verify the performance improvement from the risk-aware strategy, we conducted ablation studies by incrementally removing key components—risk-adaptive weights, the risk-aware mechanism, and the dynamics model—from RAPO.

Results on halfcheetah-medium-replay (Fig. 2) show that removing risk-adaptive weights degrades performance due to manual tuning. Removing the risk-aware mechanism causes severe performance drop and instability, confirming its role in preventing learning from high-risk data. Removing only the dynamics model reduces performance while preserving stability, indicating that model errors introduce harmful bias and justifying the need for risk awareness. Overall, the ablation study confirms that both the risk-aware strategy and adaptive weights are essential for robust offline RL.

C. Analysis of Rollout Length and Risk-Adaptive Weight(RQ.3)

While our risk-adaptive weighting reduces manual tuning, several key parameters may still affect performance. We analyze these hyperparameters to better understand RAPO.

TABLE I
PERFORMANCE COMPARISON OF VARIOUS METHODS ACROSS DIFFERENT ENVIRONMENTS AND DATASETS.

Dataset	Environment	Method							
		BC	CQL	TD3+BC	SYNTHER	PGD	MBPO	MOPO	RAPO
Random	Halfcheetah	2.6	31.4	11.0	12.2±1.1	12.9±0.9	27.8±2.8	29.4±2.7	**32.4±2.6**
	Hopper	4.1	10.9	8.5	14.6±9.4	19.5±11.2	16.9±0.7	6.9±0.7	**20.4±0.7**
	Walker	1.2	7.0	1.6	2.3±1.9	4.3±1.7	3.8±0.5	8.8±1.3	**22.8±1.9**
Medium	Halfcheetah	42.0	44.4	48.3	49.9±1.2	53.3±0.4	59.9±12.7	61.6±3.4	**62.3±2.9**
	Hopper	56.2	79.2	58.3	63.4±4.2	65.1±4.7	26.5±0.9	17.8±0.6	**67.9±4.5**
	Walker	71.0	73.3	83.7	84.8±1.4	**87.2±1.2**	7.4±2.3	52.7±3.7	71.5±6.3
Med-Rep	Halfcheetah	36.4	46.2	44.6	42.7±1.0	57.4±1.1	51.8±10.6	59.1±4.1	**62.2±3.7**
	Hopper	21.8	48.5	60.9	48.1±7.5	54.9±7.3	41.2±7.4	58.7±7.3	**76.6±3.7**
	Walker	24.9	56.7	81.8	73.7±6.5	81.7±9.1	41.7±5.8	94.1±11.5	**94.7±11.3**
Med-Exp	Halfcheetah	59.6	75.6	90.7	87.2±11.1	**100.1±2.7**	39.5±2.8	61.3±2.4	98.6±11.0
	Hopper	51.7	98.7	99.9	105.4±9.7	109.7±4.1	100.9±2.1	106.8±6.0	**111.8±5.7**
	Walker	101.2	**111.0**	110.1	110.2±0.5	108.2±0.9	-0.2±0.3	22.2±1.4	98.1±0.9
Total		472.7	682.9	699.4	694.5	754.3	417.2	579.4	**819.3**

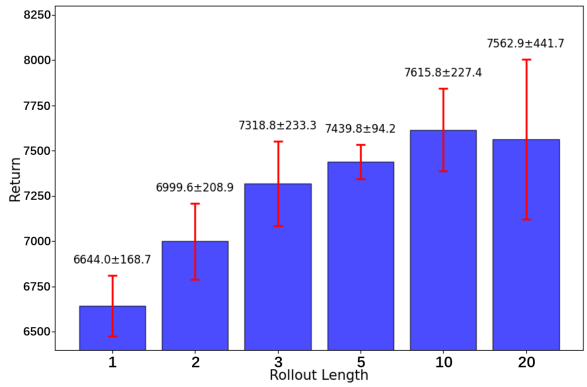

Fig. 3. **Impact of the Rollout Length h on Algorithm Performance.**

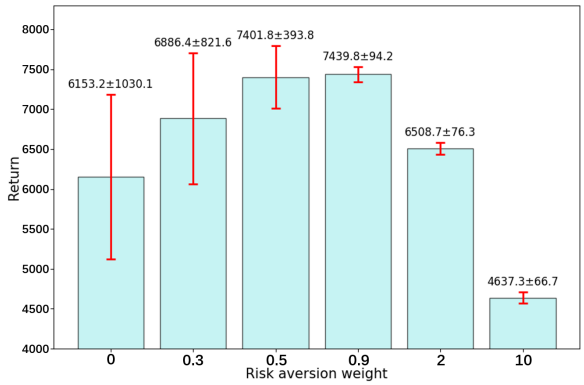

Fig. 4. **Impact of the Risk Aversion Weight β on Algorithm Performance.**

A recognized limitation of model-based offline RL is the accumulation of state prediction error over long horizons. We thus evaluate how rollout length h affects performance, as shown in Fig. 3. For $h \leq 10$, performance improves steadily with longer rollouts. However, at $h = 20$, both performance and stability decline markedly compared to $h = 10$, confirming that excessively long rollouts impair effectiveness. Thanks to the built-in risk-aware mechanism, performance degradation remains controlled even under these conditions.

Although RAPO automatically adjusts risk sensitivity, we further investigate the risk weighting mechanism by replacing the adaptive weight with a fixed **hyperparameter(risk aversion weight** β**)**. Results in Fig. 4 show that for $\beta \leq 0.9$, performance and stability improve progressively. Beyond $\beta > 0.9$, performance drops substantially despite continued stability gains—indicating that overly conservative policies sacrifice reward in pursuit of safety.

V. RELATED WORK

Reinforcement learning (RL) learns policies through online trial-and-error [21], but suffers from sample inefficiency [22]. Offline RL (batch RL) addresses this by learning from fixed datasets [23], yet faces distribution shift, causing unstable learning and divergence [4]. Conservative policy methods and data augmentation have been proposed to mitigate this.

Conservative Policy Methods. These methods constrain policy actions to prevent learning from out-of-distribution data [24]. BCQ [25] restricts updates to dataset actions, TD3+BC [17] adds a BC loss to TD3, and CQL [5] penalizes Q-values of OOD actions. While effective, they limit exploration and sacrifice performance [26].

Data Augmentation Methods. Some methods expand the offline dataset to address distribution shift [27]. SYNTHER [18] uses a diffusion model for augmentation, while PGD [19] incorporates policy gradients into denoising to generate policy-aligned data, but performance gains remain limited.

Model-Based Methods. These methods combine implicit regularization with learned dynamics models for policy interaction . MOPO [12] uses an ensemble model and constructs a conservative reward based on uncertainty. However, coupling uncertainty with reward lacks interpretability, limits control over exploration-conservatism trade-off, and relies on manually tuned penalty coefficients.

VI. CONCLUSION

To address distributional shift in offline RL and balance exploration with conservatism, we introduce a risk-aware

mechanism that quantifies long-term decision risks using an ensemble model. A riskQ network, analogous to the Q-network, estimates these risks to explicitly guide the exploration–conservatism trade-off. To eliminate hyperparameter reliance, we propose a risk-adaptive weight that adjusts risk aversion strength based on policy exploration and dataset quality. We provide theoretical analysis of the algorithm's performance lower bound and empirical validation of its effectiveness. Future work will explore this risk-aware mechanism in complex environments and integrate it with advanced RL techniques to improve policy performance and generalization, offering new directions for offline RL.

REFERENCES

[1] S. Fujimoto and S. S. Gu, "A minimalist approach to offline reinforcement learning," *Advances in neural information processing systems*, vol. 34, pp. 20132–20145, 2021.

[2] Z. Liu, J. Tian, Q. Cai, X. Zhao, J. Gao, S. Liu, D. Chen, T. He, D. Zheng, P. Jiang, *et al.*, "Multi-task recommendations with reinforcement learning," in *Proceedings of the ACM web conference 2023*, pp. 1273–1282, 2023.

[3] A. L. Chandra, I. Nematollahi, C. Huang, T. Welschehold, W. Burgard, and A. Valada, "Diwa: Diffusion policy adaptation with world models," *arXiv preprint arXiv:2508.03645*, 2025.

[4] A. Kumar, J. Fu, M. Soh, G. Tucker, and S. Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction," *Advances in neural information processing systems*, vol. 32, 2019.

[5] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in neural information processing systems*, vol. 33, pp. 1179–1191, 2020.

[6] J. Lyu, X. Ma, X. Li, and Z. Lu, "Mildly conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1711–1724, 2022.

[7] Y. Liu and M. Hofert, "Implicit and explicit policy constraints for offline reinforcement learning," in *Causal Learning and Reasoning*, pp. 499–513, PMLR, 2024.

[8] H. Xu, L. Jiang, J. Li, Z. Yang, Z. Wang, V. W. K. Chan, and X. Zhan, "Offline rl with no ood actions: In-sample learning via implicit value regularization," *arXiv preprint arXiv:2303.15810*, 2023.

[9] D. Cho, D. Shim, and H. J. Kim, "S2p: State-conditioned image synthesis for data augmentation in offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 11534–11546, 2022.

[10] J. Wei, X. Xu, Y. Lan, and T. Liu, "Adaptive data augmentation transformer for offline meta-reinforcement learning," in *Proceedings of the 2025 International Conference on Machine Learning and Neural Networks*, pp. 70–77, 2025.

[11] M. Rigter, B. Lacerda, and N. Hawes, "Rambo-rl: Robust adversarial model-based offline reinforcement learning," *Advances in neural information processing systems*, vol. 35, pp. 16082–16097, 2022.

[12] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma, "Mopo: Model-based offline policy optimization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14129–14142, 2020.

[13] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine, "Planning with diffusion for flexible behavior synthesis," *arXiv preprint arXiv:2205.09991*, 2022.

[14] H. Hasselt, "Double q-learning," *Advances in neural information processing systems*, vol. 23, 2010.

[15] O. Anschel, N. Baram, and N. Shimkin, "Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning," in *International conference on machine learning*, pp. 176–185, PMLR, 2017.

[16] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," *Cornell University - arXiv,Cornell University - arXiv*, Apr 2020.

[17] S. Fujimoto and S. Gu, "A minimalist approach to offline reinforcement learning," *Cornell University - arXiv,Cornell University - arXiv*, Jun 2021.

[18] C. Lu, P. Ball, and J. Parker-Holder, "Synthetic experience replay," Mar 2023.

[19] M. T. Jackson, M. T. Matthews, C. Lu, B. Ellis, S. Whiteson, and J. Foerster, "Policy-guided diffusion," *arXiv preprint arXiv:2404.06356*, 2024.

[20] M. Janner, J. Fu, M. Zhang, and S. Levine, "When to trust your model: Model-based policy optimization," *Advances in neural information processing systems*, vol. 32, 2019.

[21] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man, and Cybernetics*, p. 834–846, Sep 1983.

[22] S. Wang, N. Si, J. Blanchet, and Z. Zhou, "Sample complexity of variance-reduced distributionally robust q-learning," *Journal of Machine Learning Research*, vol. 25, no. 341, pp. 1–77, 2024.

[23] S. Lange, T. Gabel, and M. Riedmiller, *Batch Reinforcement Learning*, p. 45–73. Jan 2012.

[24] D. Wang, L. Li, W. Wei, Q. Yu, J. Hao, and J. Liang, "Improving generalization in offline reinforcement learning via adversarial data splitting," in *Forty-first International Conference on Machine Learning*, 2024.

[25] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," *arXiv: Learning,arXiv: Learning*, Dec 2018.

[26] Y. Mao, H. Zhang, C. Chen, Y. Xu, and X. Ji, "Supported trust region optimization for offline reinforcement learning," in *International Conference on Machine Learning*, pp. 23829–23851, PMLR, 2023.

[27] C. Pan, Z. Yi, G. Shi, and G. Qu, "Model-based diffusion for trajectory optimization," *Advances in Neural Information Processing Systems*, vol. 37, pp. 57914–57943, 2024.